A Case for Bufferless Routing in On-Chip Networks

Resumido por Luciano Jerez Chaves - RA079759

Moscibroda, T. and Mutlu, O. 2009. A Case for Bufferless Routing in On-Chip Networks. *In Proc. of the 36th International Symposium on Computer Architecture* (ISCA '09). ACM, New York, NY, USA, 196-207.

O artigo em questão discute acerca do processo de roteamento de mensagens nas redes de comunicação que interligam os núcleos e as memórias *caches* dos modernos *chips* multiprocessados. No modelo atual, é associado um roteador à cada núcleo do processador. Estes roteadores se conectam com alguns de seus roteadores vizinhos, formando uma rede de comunicação de múltiplos saltos onde uma mensagem enviada por alguma origem pode passar por vários outros roteadores antes de chegar ao seu destino. Quando uma mensagem chega em um roteador, ela é armazenada em um *buffer* até que seja analisada e encaminhada para o enlace de saída apropriado. O problema levantado pelos autores do artigo diz respeito ao consumo energético e ao espaço ocupado pelos *buffers* nestes roteadores. Para melhorar estes aspectos, os autores propõem uma nova abordagem que dispensa o uso dos *buffers* no processo de roteamento das mensagens.

Para utilizar a solução proposta, existem duas restrições em relação ao projeto da rede: (1) a quantidade de enlaces de entrada em cada roteador tem que ser no máximo igual a quantidade de enlaces de saída neste roteador; e (2) qualquer roteador tem que ser alcançável a partir de todos os outros, formando um grafo conexo.

Para eliminar os *buffers* dos roteadores, é preciso encaminhar todas as mensagens (ou *flits*[1]) recebidos nos enlaces de entrada para algum dos enlaces de saída, mesmo que o enlace escolhido não seja o que aproxima a mensagem de seu destino final. Essa abordagem funciona por conta da primeira restrição em relação o número de enlaces nos roteadores (nenhuma mensagem deixará de ser encaminhada). O algoritmo parte do princípio que as mensagens desviadas eventualmente irão alcançar o seu destino (segunda restrição). Dessa forma, os *links* se tornam os *buffers* da rede, diminuindo a vazão máxima alcançável. A abordagem apresentada parece audaciosa para redes em geral, mas no caso específico das redes internas dos processadores, onde a carga pode ser considerada baixa, é possível conseguir benefícios com o uso desta técnica ao custo de algum incremento no tempo gasto para a entrega das mensagens (latência).

Para realizar a escolha de qual mensagem será transferida para qual enlace, o roteador constrói uma fila de prioridades com todas as mensagens recebidas. Cada mensagem é retirado da fila e enviada pelo enlace de maior interesse da mensagem, desde que este enlace ainda esteja disponível. Caso contrário, é utilizado o próximo enlace de maior interesse que esteja disponível. Para organizar as mensagens por prioridade, utiliza-se a abordagem do "mais velho primeiro", garantindo a menor latência e evitando *deadlocks* e *livelocks* (já que toda mensagem em algum momento se tornará a mais velha na rede e será roteada pelo caminho correto). Este processo também pode ser utilizada por roteadores que possuem *buffer*, desde que as restrições apresentadas sejam atendidas.

Para o caso da fragmentação das mensagens, os autores propõem o uso de uma solução semelhante ao roteamento *wormhole*, onde os demais fragmentos são sempre encaminhados pelo mesmo enlace do primeiro, de maneira a não provocar variações na latência de chegada de cada fragmento no destino. Entretanto, o artigo propõe uma "relaxação" na proposta original, podendo truncar a sequência de *flits* sempre que um *flit* "mais velho" precisa usar o mesmo enlace de saída, de maneira a garantir a ausência dos *livelocks*. Essa abordagem demanda por um *buffer* maior no receptor, que terá que lidar com os *flits* que podem ser recebidos fora de ordem por conta da truncagem da sequência.

Após a apresentação e descrição do algoritmo proposto, os autores discutem acerca da implementação deste algoritmo dentro do *pipeline* de roteamento. Uma grande vantagem em relação as soluções existentes é que o método proposto permite reduzir de 3 para 2 o número de estágios do *pipeline*, podendo chegar a 1 com o custo de um barramento adicional entre os roteadores para trafegar as informações sobre as rotas das mensagens.

Para avaliar o desempenho da técnica proposta, foi utilizado um simulador de redes acoplado à um emulador `x86`, que executou aplicações do *benchmark* SPEC CPU2006 em diferentes configurações de redes de interconexão. Também foram simuladas cargas de tráfego sintético. Foram consideradas a latência na entrega dos *flits*, a vazão de saturação da rede, a demanda por *buffers* no receptor, e o consumo energético da rede. Além disso, também foi avaliado o desempenho das aplicações executando sobre a rede com a arquitetura proposta, utilizando o *speedup* ponderado em relação o `IPC` alcançado por cada tipo de rede como métrica de qualidade.

No geral, o uso da técnica proposta conseguiu reduzir o consumo energético entre 40% e 45%, com uma queda no desempenho do sistema de apenas 1.5% nas configurações reais consideradas pouco estressadas. Este valor pode chegar à 17% em situações atípicas. Com o uso de tráfegos sintéticos, foi possível avaliar a vazão de saturação da rede, que foi reduzida entre 20% e 35% da vazão de saturação normal (roteamento com uso de *buffers*). A abordagem proposta demanda por *buffers* até 25% maiores nos receptores, mas ainda consegue reduzir a área total utilizada no circuito em até 60%.

Para concluir o artigo, os autores apresentam uma rápida revisão dos algoritmos de roteamento existentes para estas redes de interconexão, e reforçam as inovações existentes neste trabalho.

[1] Quando uma mensagem é muito longa, ela é dividida em partes menores, denominadas *flits*.